NAVAL POSTGRADUATE SCHOOL

MONTEREY, CALIFORNIA

THESIS

INFORMATION SHARING FOR COMPUTING TRUST METRICS ON COTS ELECTRONIC COMPONENTS

by

William J. McMillon

September 2008

Thesis Advisors: James B. Michael
Raymond Buettner

Approved for public release; distribution is unlimited

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE *Form Approved OMB No. 0704-0188*

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instruction, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden, to Washington headquarters Services, Directorate for Information Operations and Reports, 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302, and to the Office of Management and Budget, Paperwork Reduction Project (0704-0188) Washington DC 20503.

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2008	**3. REPORT TYPE AND DATES COVERED** Master’s Thesis

4. TITLE AND SUBTITLE Information Sharing for Computing Trust Metrics on COTS Electronic Components

5. FUNDING NUMBERS

6. AUTHOR(S) LT William J. McMillon, USN

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
Naval Postgraduate School
Monterey, CA 93943-5000

8. PERFORMING ORGANIZATION REPORT NUMBER

9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES)
N/A

10. SPONSORING/MONITORING AGENCY REPORT NUMBER

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government.

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Approved for public release; distribution unlimited

12b. DISTRIBUTION CODE
A

13. ABSTRACT (maximum 200 words)

The Department of Defense (DoD) has become highly reliant on commercial-off-the-shelf (COTS) technology in mission-critical unclassified systems to reduce both the cost time to acquire a system, and standardize support for deployed systems. It is challenging for the DoD to determine whether and how much to trust in COTS components, given uncertainty and incomplete information about the developers and suppliers of COTS components as well as the capabilities provided by COTS components.

The purpose of this thesis is to explore the current landscape of DoD information assurance (IA) as it pertains to COTS components, show how Jøsang's trust model can be used to calculate trust based on opinions provided by multiple government and non-government services, and explore the need for cross-domain sharing of information to support populating, maintaining, and using the trust models.

14. SUBJECT TERMS Commercial off the Shelf, COTS, Information Sharing, Information Assurance

15. NUMBER OF PAGES
67

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

THIS PAGE INTENTIONALLY LEFT BLANK

INFORMATION SHARING FOR COMPUTING TRUST METRICS ON COTS ELECTRONIC COMPONENTS

William J. McMillon
Lieutenant, United States Navy
B.S., University of Texas at Austin, 2001

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN INFORMATION WARFARE SYSTEMS ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL
September 2008

Author: William J. McMillon

Approved by: James B. Michael
Thesis Advisor

Raymond R. Buettner
Co-Advisor

Daniel C. Boger
Chairman, Department of Information Sciences

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

The Department of Defense (DoD) has become highly reliant on commercial-off-the-shelf (COTS) technology in mission-critical unclassified systems to reduce both the cost time to acquire a system, and standardize support for deployed systems. It is challenging for the DoD to determine whether and how much to trust in COTS components, given uncertainty and incomplete information about the developers and suppliers of COTS components as well as the capabilities provided by COTS components.

The purpose of this thesis is to explore the current landscape of DoD information assurance (IA) as it pertains to COTS components, show how Jøsang's trust model can be used to calculate trust based on opinions provided by multiple government and non-government services, and explore the need for cross-domain sharing of information to support populating, maintaining, and using the trust models.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

BIOS	Basic Input Output System
COTS	Commercial Off the Shelf
C&A	Certification and Accreditation
DoD	Department of Defense
DoN	Department of the Navy
IA	Information Assurance
IB	Information Broker
IV&V	Independent Verification and Validation
GPS	Global Positioning Satellite
MLS	Multi-level secure
NIST	National Institute of Standards and Technology
NDI	Non Developmental Item
NGO	Non-governmental organization
NSA	National Security Agency
OS	Operating System
PC	Personal Computer
PKI	Public Key Infrastructure
PPS	Precise Positioning Service
RAM	Random Access Memory
RBAC	Role Based Access Control
SA	Selective Availability
SDK	Software Development Kit
SDLC	Systems Development Life Cycle
TDC	Trusted Database Connector
USB	Universal Serial Bus

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

The author would like to thank Bret Michael and Raymond Buettner for their guidance in developing this thesis and their editorial skills. Additionally, I would like to thank my wife Sharon for offering words of encouragement when appropriate, and throwing me out of the house when necessary in order to get thesis work done.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

This thesis identifies challenges in the use of commercial-off-the-shelf (COTS) components, a type of non-developmental items (NDI), in mission-critical but unclassified systems. The U.S. Department of Defense (DoD) increasingly turns to acquiring COTS components in order to reduce both the cost and time to acquire a system, and to standardize support for deployed systems.

On June 29, 1994, then Secretary of Defense William Perry issued a memorandum prohibiting the use of legacy military defense standards without a waiver and encouraged the use of industry standards. Weapon systems were required to use "performance specifications" that described the desired features of the weapon system instead of citing military standards [1]. This memo removed the requirement of military standards and specifications and has lead to DoD's present-day reliance on COTS components.

The capabilities provided by COTS components do not always match up with the requirements of DoD. Usually, provision of built-in security features for such mass-marketed components is not the highest priority of the developer. Instead the developer's focus is on time-to-market concerns and the primary functionality of the system [2]. The reliability of such components and the security of such systems are assigned a low priority relative to the functionality of the system.

The typical development cycle for software and electronic components is now less than one year, making it difficult for DoD entities to perform extensive independent verification and validation (IV&V), as well as Information Assurance (IA) certification and accreditation (C&A), on the systems containing COTS components.

COTS components tend to have a short life cycle, contributing to the challenge for the DoD in maintaining legacy systems due to the unavailability of components. "Component churn," that is the movement of components into or out of service, makes IV&V and IA C&A even more challenging.

A trustworthy system is one that provides the appropriate levels of correctness and robustness in accomplishing its mission [3]. Trust in this context is especially important in critical yet unclassified systems which affect lives and national assets, yet the systems have not been vetted to the same extent as systems that process classified data.

The purpose of this thesis is to explore the current landscape of DoD IA as it pertains to COTS components, show how Jøsang's trust model can be used to calculate trust based on opinions provided by multiple government and non-government services, and explore the need for cross-domain sharing of information to support populating, maintaining, and using the trust models.

A. MOTIVATIONS FOR THIS THESIS

Consider the reverse engineering efforts undertaken by large communities of people to exploit commercially available hardware such as video game consoles, cell phones, and Global Positioning System (GPS) receivers, in order to remove installed security features and add functionality not included in the original systems. Many of these groups receive little or no monetary compensation for their efforts and are only motivated by the notoriety they receive from their exploits. Such groups with little to no sponsorship have successfully thwarted security systems specifically designed to prevent such actions.

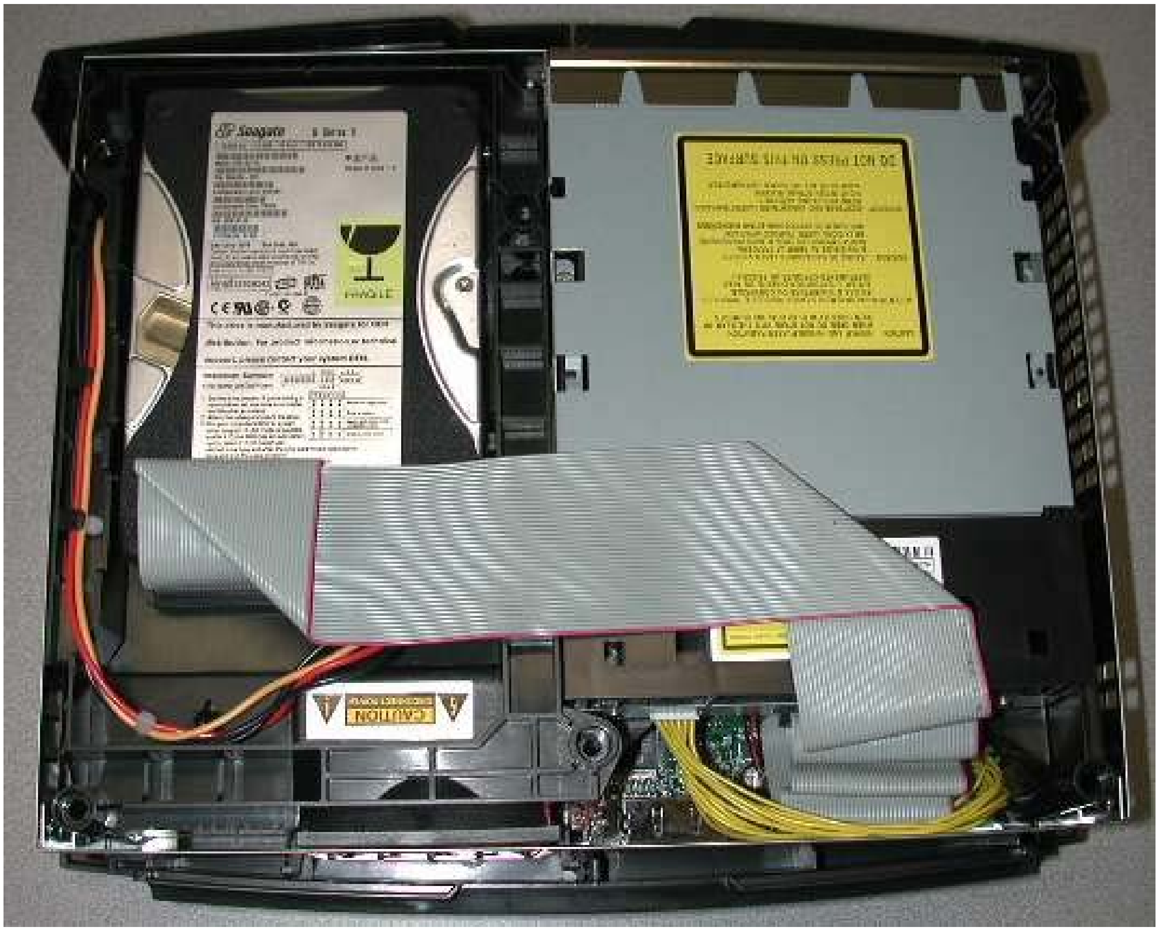


Figure 1. Photograph of Xbox internals.

For example, in November of 2001, Microsoft released the Xbox video game console based on common personal computer (PC) hardware. The Xbox is essentially a PC with an Intel Mobile Celeron processor, hard drive, nVidia GeForce video card, random access memory (RAM), Ethernet port, and Universal Serial Bus (USB) ports cleverly disguised as game controller ports [4]. The resemblance of the Xbox to a common PC. On the left is the hard disk drive and on the right is the DVD drive. It is thought that Microsoft chose common off-the-shelf components to reduce cost, development time, and time-to-market. Because of the use of such common components, the aforementioned groups were able to relatively quickly exploit the system since the groups were quite familiar with how PCs and their peripheral devices operated. Andrew Huang, at that time a doctoral student at MIT, is credited with extracting the Xbox basic input/output system (BIOS) and publishing it on his website. Eventually he was able to intercept the RC4 encryption key used to encrypt the bootloader and BIOS by monitoring traffic on the HyperTransport bus. The bootloader and BIOS were then modified by various groups to allow the Xbox to boot executables without the correct RSA signature or boot from an unapproved media (e.g., boot from the Xbox hard drive vice the DVD drive). The altered BIOS and bootloader prompted the widespread piracy of Xbox games. Additionally, the leaking of Microsoft's official Software Development Kit (SDK) allowed the development of various "homebrew" applications and the porting of an assortment of applications. It is possible to install the Linux operating system (OS) on the Xbox hard drive and use the USB controller ports to add peripherals such as a keyboard and mouse [5].

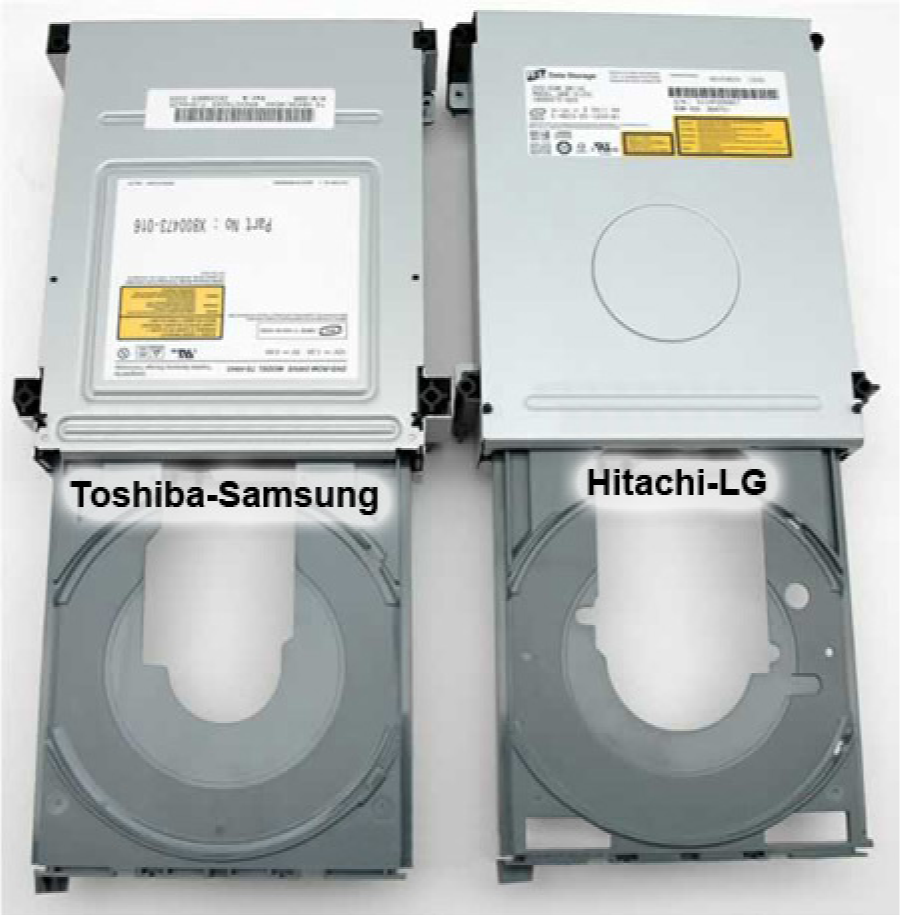


Figure 2. Photograph used to identify Xbox 360 DVD drives.

The successor to the Xbox, the Xbox 360, was released in November of 2005. Microsoft hardened the internal OS of the newer system, but still employed common, commercially available, DVD readers in its product. Two of the many types of DVD drives utilized are illustrated in Figure 2. Eventually, most Xbox 360 DVD drives' firmware were reverse engineered and altered to report all media inserted into the drives as authenticated. Similar reverse engineering efforts have allowed the execution of unauthenticated code on products such as Sony's Play Station Portable, Apple's Iphone, and various GPS receivers running the Windows CE OS.

The implications of such actions are clear. Unfunded groups with limited resources are able to remove security features of commercial products designed to deter such actions. A well-funded adversary, such as state-sponsored information warriors or non-state actors such as members of organized crime syndicates or terrorist organizations, may be able to exploit DoD's reliance on COTS in much the same way or perhaps by interfering with the design, development, and implementation of COTS components. DoD must examine the issues surrounding its dependence on COTS and if appropriate implement more stringent acquisition policies.

It is not enough to vet just the components of systems. It is also necessary to scrutinize the developers or suppliers of the components. The behavior of a system containing two or more components must be understood too.

II. BACKGROUND

A. INFORMATION ASSURANCE REQUIREMENTS FOR COMMERICIAL OFF THE SHELF TECHNOLOGY

Information Assurance (IA) is defined as: "Measures that protect and defend information systems by ensuring their availability, integrity, authentication, confidentiality, and non-repudiation…" by the Committee on National Security Systems [6]. IA provides a measure of confidence that a particular software or hardware system will perform as designed and has not been tampered with or compromised. In order to have confidence in the system, we must first have confidence in all of the components of that system.

The Computer Security Act of 1987 clarified the definition of "national security-related information," and assigned responsibility of all federal unclassified information systems (including DoD systems) to the National Institute of Standards and Technology (NIST).

However, all "national security-related information" systems are governed by the National Security Agency (NSA). Therefore, all non-"national security-related information" COTS IT systems must meet NIST's IA requirements.

B. IA VULNERABILITIES THROUGHOUT THE ENTIRE LIFE CYCLE

Vulnerabilities capable of negatively affecting IA can be introduced anywhere in the product life cycle. Potential vulnerabilities in the Systems Development Life Cycle (SDLC), vulnerabilities during implementation, vulnerabilities during production and manufacturing, and vulnerabilities during distribution will be discussed in this chapter.

Infiltrating a component manufacturer's development system and allowing unsuspecting users to install components for malactors is becoming more efficient for the malactors than attacking each installation individually. In many cases, infiltrating the development cycle presents the weakest-link in the component life cycle [7].

1. Systems Development Life Cycle

The SDLC refers to the phases of development of a system. There are many well-known SDLC models, the most popular of which are:

- Waterfall
- V-shaped
- Spiral
- Agile

a. Standard SDLCs

In this context, the term "standard" will refer to the un-modified or academic definition of each of the SDLCs. Many SDLCs have been modified to fit a particular use, or to fit a specific timeline.

b. The Waterfall Model

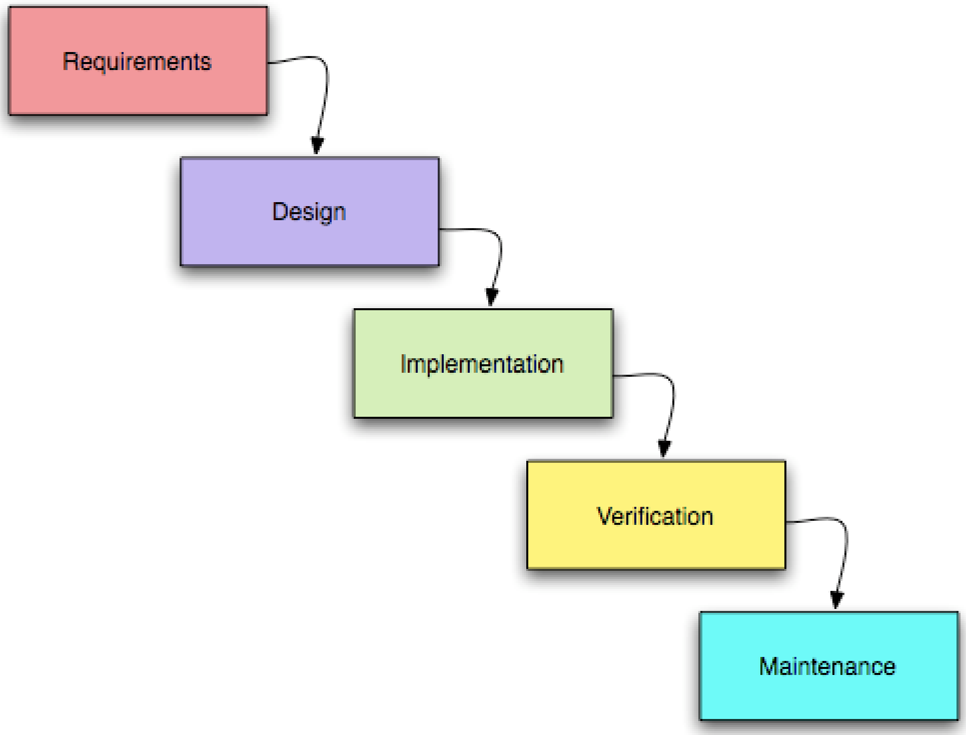


Figure 3. Un-modified "Waterfall" model. Work proceeds from the top phase and cascades downward.

As illustrated in the above figure, the waterfall flows from one phase of the process to another. Each phase is completed sequentially, and one phase is not started until the previous one is both complete and verified. The phases include:

- Requirements: The systems specifications are established to include constraints and goals, usually by analyzing the needs of the users.
- System design: Divides the requirements into either hardware or software as appropriate and establishes an overall system architecture. Determines a framework by which requirements can be implemented. Includes user interface, data structures, etc.

- Implementation: Each component of the hardware or software is realized and tested to ensure it meets the specification.
- Integration or installation: Individual components are integrated or installed. Testing is performed on the entire system to ensure software requirements have been met.
- Operation and maintenance: Usually the longest and most expensive phase. The system is put into use. Maintenance is required for undiscovered deficiencies [8].

The "Waterfall" model is easy to use and provides a rigid structure to the developmental process. Milestones are easy to discern and track. However, it can be argued that it is difficult if not impossible to implement this model because of the difficulty in completely finishing one phase before moving on to the next. Additionally if the requirements and system design phases were not correctly completed, it may be impossible to continue to implement the system. Following the Waterfall model makes it difficult to modify security or IA requirements in later phases. Even with these possible flaws, this model (and variations of it) is often used for acquisition of systems in which quality is more important than cost or schedule.

c. V-shaped Model

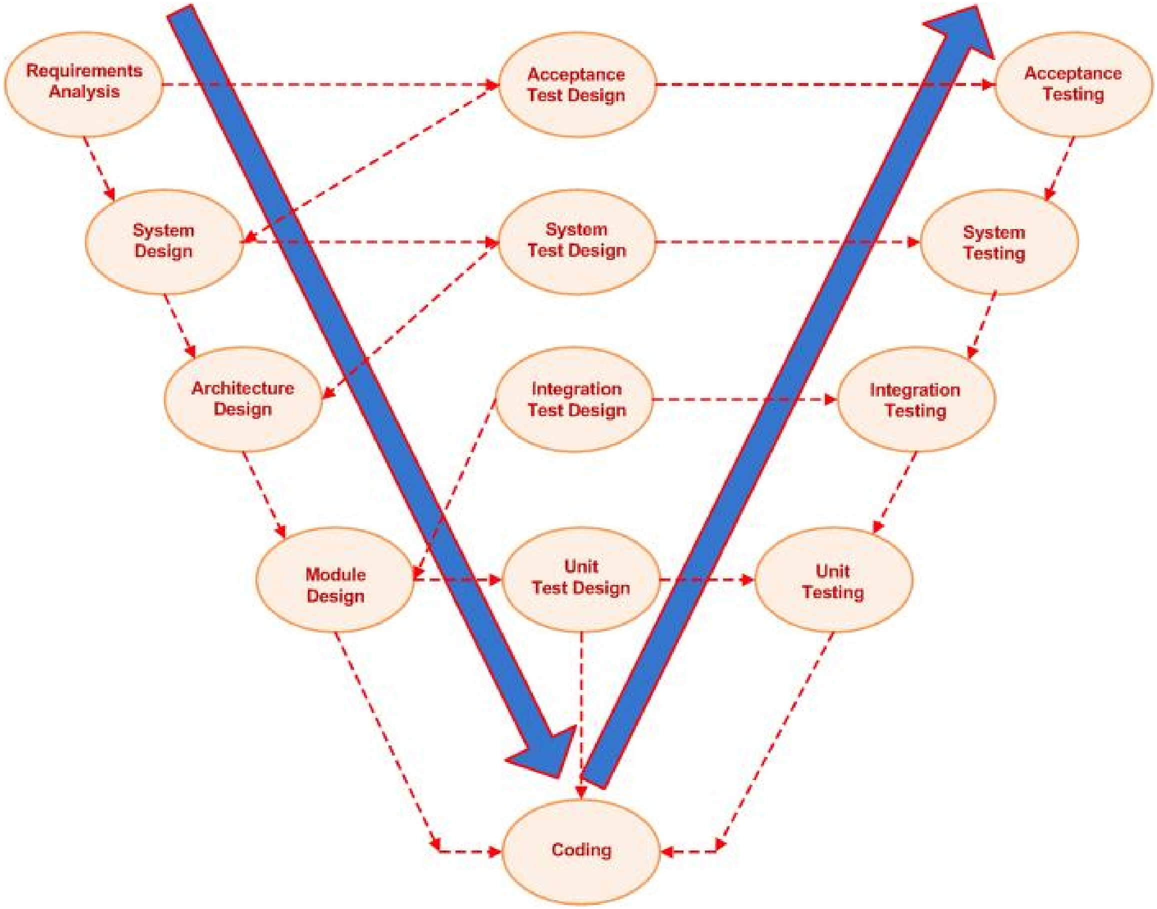


Figure 4. V-Shaped model.

The V-Shaped model is a variant of the Waterfall model that puts emphasis on verification and validation (V&V) of the system. The model ties each phase of development to a phase of testing

The V-Shaped model has the same strengths and weakness as with the Waterfall but is more suited to systems that require V&V of the system early and often throughout development. Additionally, like the Waterfall method, it does not easily allow for changing requirements or concurrent events.

d. Spiral Development

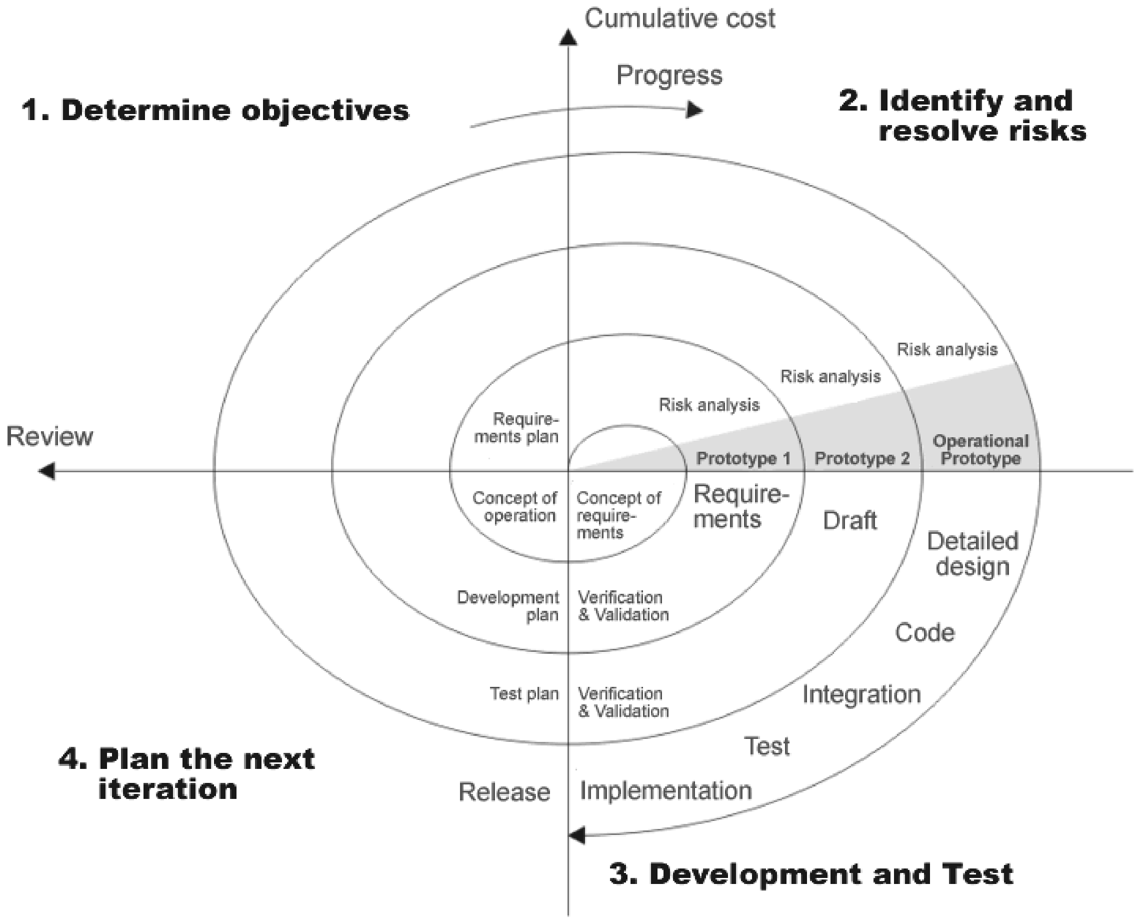


Figure 5. Spiral Model (Boehm 1988).

The Spiral model was proposed by Barry Boehm [9]. It is an iterative process where the same four steps are carried out for each phase of the previously discussed models. For example, the innermost loop might represent requirements gathering and the next loop system design, etc. What truly distinguishes the Spiral model from other models is that the model calls for analyzing risk in every phase of development.

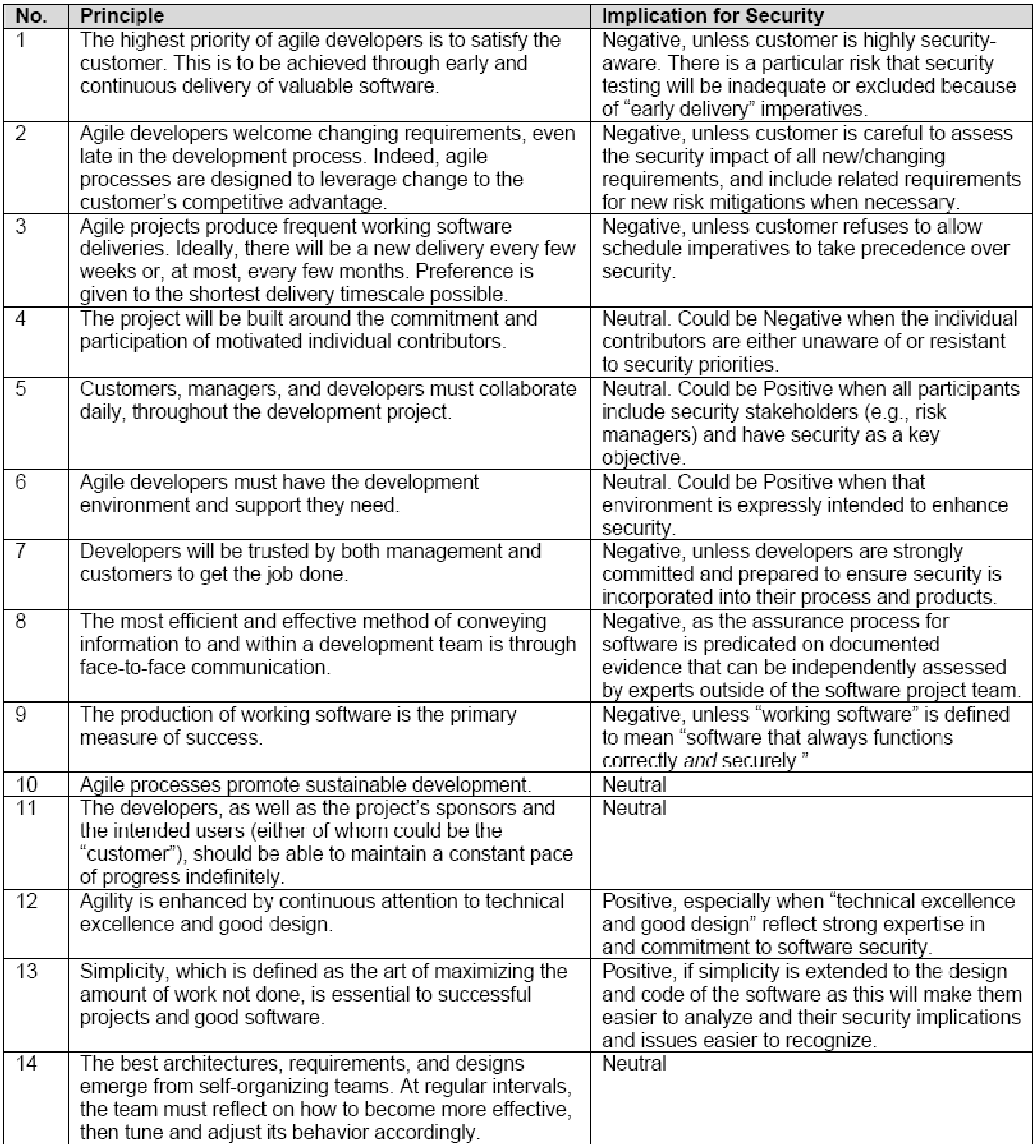
The Spiral model provides an indication of insurmountable risks early in the process because high-risk functions are often developed first. The model promotes the management of recognized risks prior to attempting traditional phased software development. The Spiral model is appropriate when costs and risk evaluation are important and when a prototype is needed. The Spiral model may be unsuitable for smaller or lower risk projects.

e. Compressed SDLCs

> Pressure to be first to market and retain what is known as *mind share* compresses the development cycle so much that software engineering methods are often thrown out the window…often leaving rigorous testing to the users [10].

In general, SDLC models are compressed by overlapping stages, working various stages of the model in parallel, or both. Compressing the development cycle can lead to decreased time for testing. Decreased time for testing can lead to an increase in the number and severity of vulnerabilities in systems.

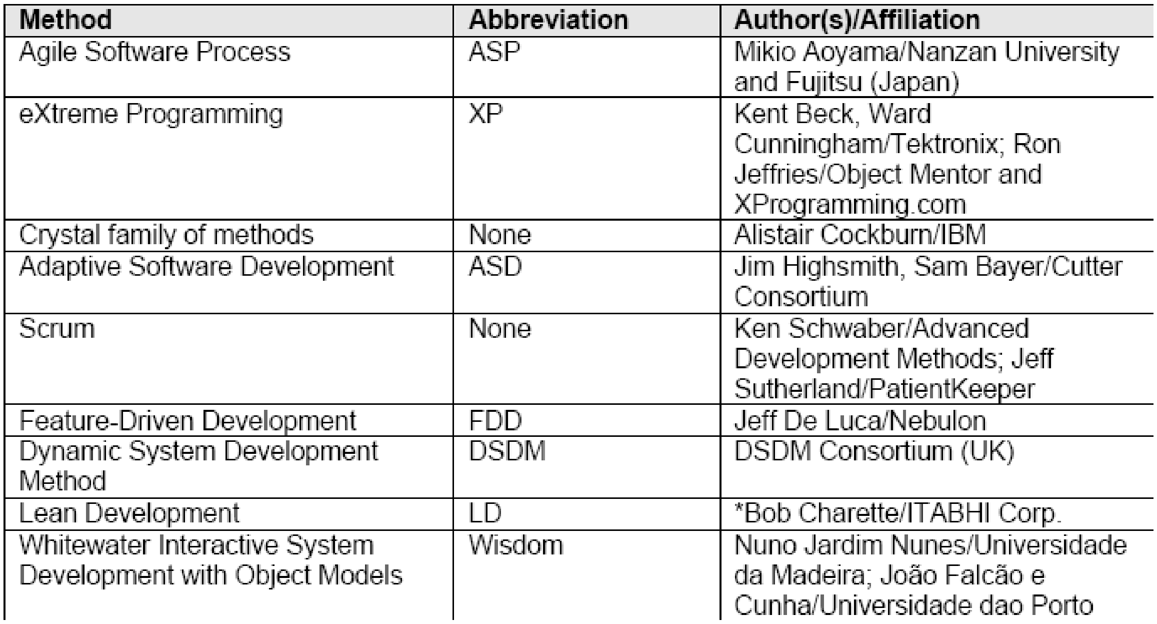
f. Agile



No.	Principle	Implication for Security
1	The highest priority of agile developers is to satisfy the customer. This is to be achieved through early and continuous delivery of valuable software.	Negative, unless customer is highly security-aware. There is a particular risk that security testing will be inadequate or excluded because of "early delivery" imperatives.
2	Agile developers welcome changing requirements, even late in the development process. Indeed, agile processes are designed to leverage change to the customer's competitive advantage.	Negative, unless customer is careful to assess the security impact of all new/changing requirements, and include related requirements for new risk mitigations when necessary.
3	Agile projects produce frequent working software deliveries. Ideally, there will be a new delivery every few weeks or, at most, every few months. Preference is given to the shortest delivery timescale possible.	Negative, unless customer refuses to allow schedule imperatives to take precedence over security.
4	The project will be built around the commitment and participation of motivated individual contributors.	Neutral. Could be Negative when the individual contributors are either unaware of or resistant to security priorities.
5	Customers, managers, and developers must collaborate daily, throughout the development project.	Neutral. Could be Positive when all participants include security stakeholders (e.g., risk managers) and have security as a key objective.
6	Agile developers must have the development environment and support they need.	Neutral. Could be Positive when that environment is expressly intended to enhance security.
7	Developers will be trusted by both management and customers to get the job done.	Negative, unless developers are strongly committed and prepared to ensure security is incorporated into their process and products.
8	The most efficient and effective method of conveying information to and within a development team is through face-to-face communication.	Negative, as the assurance process for software is predicated on documented evidence that can be independently assessed by experts outside of the software project team.
9	The production of working software is the primary measure of success.	Negative, unless "working software" is defined to mean "software that always functions correctly *and* securely."
10	Agile processes promote sustainable development.	Neutral
11	The developers, as well as the project's sponsors and the intended users (either of whom could be the "customer"), should be able to maintain a constant pace of progress indefinitely.	Neutral
12	Agility is enhanced by continuous attention to technical excellence and good design.	Positive, especially when "technical excellence and good design" reflect strong expertise in and commitment to software security.
13	Simplicity, which is defined as the art of maximizing the amount of work not done, is essential to successful projects and good software.	Positive, if simplicity is extended to the design and code of the software as this will make them easier to analyze and their security implications and issues easier to recognize.
14	The best architectures, requirements, and designs emerge from self-organizing teams. At regular intervals, the team must reflect on how to become more effective, then tune and adjust its behavior accordingly.	Neutral

Table 1. Core principles of the Agile Manifesto (From Department of Homeland Defense in *Security in the Software Lifecycle*).

The Agile model operates on the belief that it is impossible to design a system without first providing a rudimentary version of the system to users and then observing the results. "It may be only after a system is delivered and users gain experience with it that the real requirements become clear" [11]. Most Agile-based methods adhere to the Agile Manifesto whose principles and applicability to security are listed by the Department of Homeland Defense in *Security in the Software Lifecycle* shown in Table 1.



Method	Abbreviation	Author(s)/Affiliation
Agile Software Process	ASP	Mikio Aoyama/Nanzan University and Fujitsu (Japan)
eXtreme Programming	XP	Kent Beck, Ward Cunningham/Tektronix; Ron Jeffries/Object Mentor and XProgramming.com
Crystal family of methods	None	Alistair Cockburn/IBM
Adaptive Software Development	ASD	Jim Highsmith, Sam Bayer/Cutter Consortium
Scrum	None	Ken Schwaber/Advanced Development Methods; Jeff Sutherland/PatientKeeper
Feature-Driven Development	FDD	Jeff De Luca/Nebulon
Dynamic System Development Method	DSDM	DSDM Consortium (UK)
Lean Development	LD	*Bob Charette/ITABHI Corp.
Whitewater Interactive System Development with Object Models	Wisdom	Nuno Jardim Nunes/Universidade da Madeira; João Falcão e Cunha/Universidade dao Porto

Table 2. Major Agile Methods (From Department of Homeland Defense in *Security in the Software Lifecycle*).

Not every SDLC shown above explicitly takes into account security in its procedures. While each of the above SDLCs can produce secure components, better results are achieved when security is considered at the beginning and throughout the process. Retrofitting security onto the product or component (if it can be done at all) after it has been released, does not lead to a desired security state.

2. Development and Implementation Strategies

Development strategies are ways of implementing a product or service and include incremental development and evolutionary development.

a. Incremental Development

Incremental development involves pre-planned segmented development of the product or components in increments. This strategy is often selected to accommodate funding limitations, handle contractor specialties, simplify deployment plans, improve development sequence, and deal with integration issues [12].

In the incremental model, customers define an outline of the services to be provided by the system. Each of the services is then prioritized and a number of delivery increments is defined [13]. This allows for the construction of a partial implementation of a total system. As each increment is added to the total system, functionality is increased. Pieces of the total system are provided earlier so that customers can immediately benefit from the new system. This model requires well-defined module interfaces (e.g., APIs in a software system) since some parts of the system will be delivered much earlier than others. The incremental approach relies on a divide-and-conquer strategy for development.

b. Evolutionary Development

Evolutionary Development involves successive improvements of products or components based on experience with prior versions. This strategy is often selected to accommodate uncertain requirements, changing problem environments, and challenging technology objectives [14].

The evolutionary development strategy combines the requirements, design, and testing phases of system development to quickly produce a prototype for user testing from a set of vague user needs. The prototype is then evaluated by the end users and feedback is submitted. Developers take the feedback and improve on the original prototype. This model is perceived as not scaling well and is thought to produce un-organized, un-maintainable code that is difficult to reuse.

Security concerns must be taken into account when implementing either incremental or evolutionary developmental strategies. Special consideration should be given to the implementation of the evolutionary development strategy because it is likely components utilizing this strategy are first-generation and lessons learned from previous generation component installations are not available. When implementing incremental development strategies, the "lessons learned" and any other information from previous use or installation should be utilized in order to avoid making the same mistake.

3. Production and Manufacturing Vulnerabilities

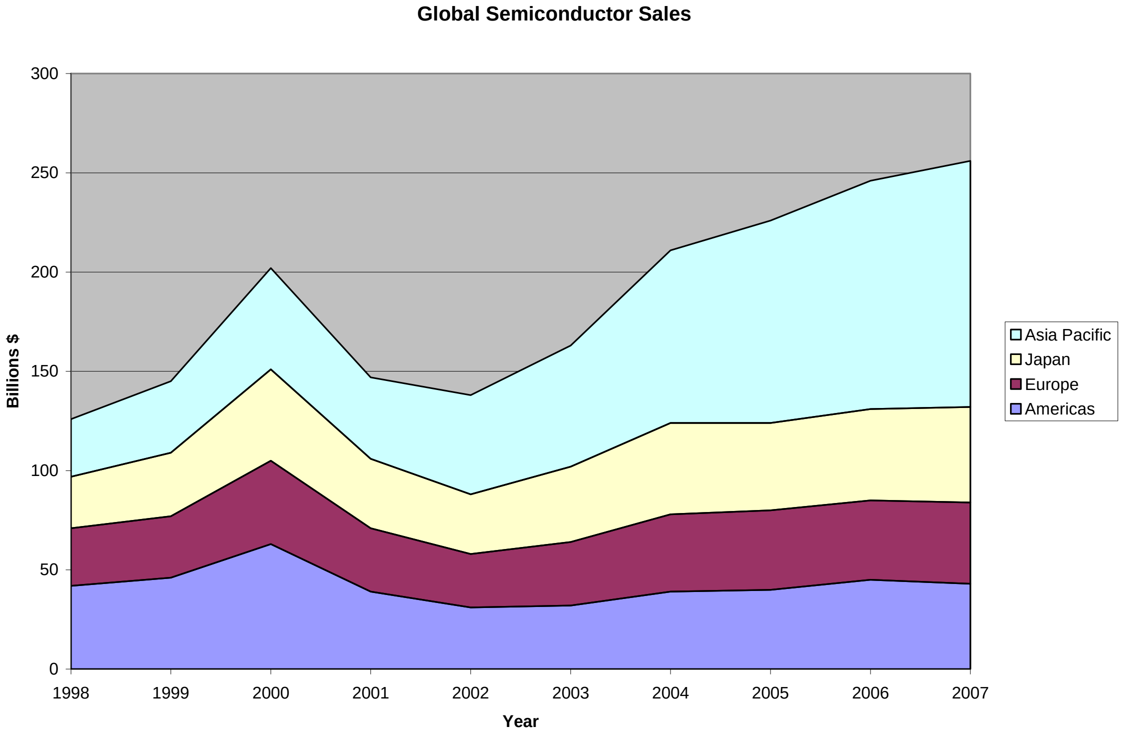


Figure 6. Global Semiconductor Sales (data from: Semiconductor Industry Association).

In the early days of the semiconductor industry, DoD and NASA represented a large percentage of the overall sales of all semiconductors and therefore could easily drive the direction of product development and dictate manufacturing requirements. In today’s global economy, DoD and NASA now represent less than one percent of the worlds semiconductor market [15]. "While the military provided the original test bed for many computers and microelectronics, defense needs are not the driver for the newest technologies in these fields in most cases” [16].

Market forces have led to the migration of the semiconductor industry from industrialized nations such as the United States, Japan, and Europe to countries in Asia where the cost of labor, land, and material are significantly lower.

This migration has led to difficulties of trust and IA with respect to these semiconductor components. As of May 2008, 49% of semiconductor manufacturing occurs in the Asia Pacific region and only 16% occurs within North and South America.

During the 18th century, British forests became depleted of Baltic fir, prized for its use in wooden ships of war. In order to fulfill the demand for quality timber, the British turned to its American colonies which had a nearly infinite supply of Live oak that was well suited for ship construction and perhaps better suited than the highly coveted Baltic fir. The American colonies began exporting large quantities of oak across the Atlantic for use by British ship builders. Soon, the British realized it would be more cost efficient to send ship builders across the Atlantic to the Americas and teach the soon to be Americans how to build ships. Eventually, the colonists became proficient at building ships, and were able to improve upon the British methods. This outsourcing eventually allowed the colonies to hoard the best timber for themselves in order to build ships such as the USS CONSITITUTION that were able to outrun many ships of the line at the time and proved invaluable during the American Revolution [17]. This anecdote can be applied to today's reliance on foreign manufacturing of COTS components. We have exported COTS component manufacturing technology overseas in an effort to be more cost efficient without taking into account the possible consequences of our actions.

4. Distribution Vulnerabilities

The networked nature of the modern world has produced unique distribution vulnerabilities with respect to both hardware and software.

a. Distribution Vulnerabilities and Software

In the past, software was distributed either by a physical medium or pre-installed on new computer systems. This method of distribution offered some assurance since it is presumed difficult to infiltrate such a closed distribution chain. However, with the popularity of the Internet, online distribution is more prevalent then ever. Even when software is distributed by physical means, it is almost always updated via the Internet. This dependence on a publicly accessible network to update software has encouraged malactors to infiltrate the software distribution supply.

On August 22, 2008, Red Hat released a statement indicating that an intruder into their network was able to get a small number of OpenSSH packages relating to its Red Hat Enterprise Linux versions 4 and 5 signed by Red Hat's private Public Key Infrastructure (PKI) key [18].

If a malactor had been able to get their altered OpenSSH signed software into one of Red Hat's many official mirrors undetected, they would have easily been able to install their software on any workstation or server using these mirrors potentially allowing them root access to thousands of machines.

b. Distribution Vulnerabilities and Hardware

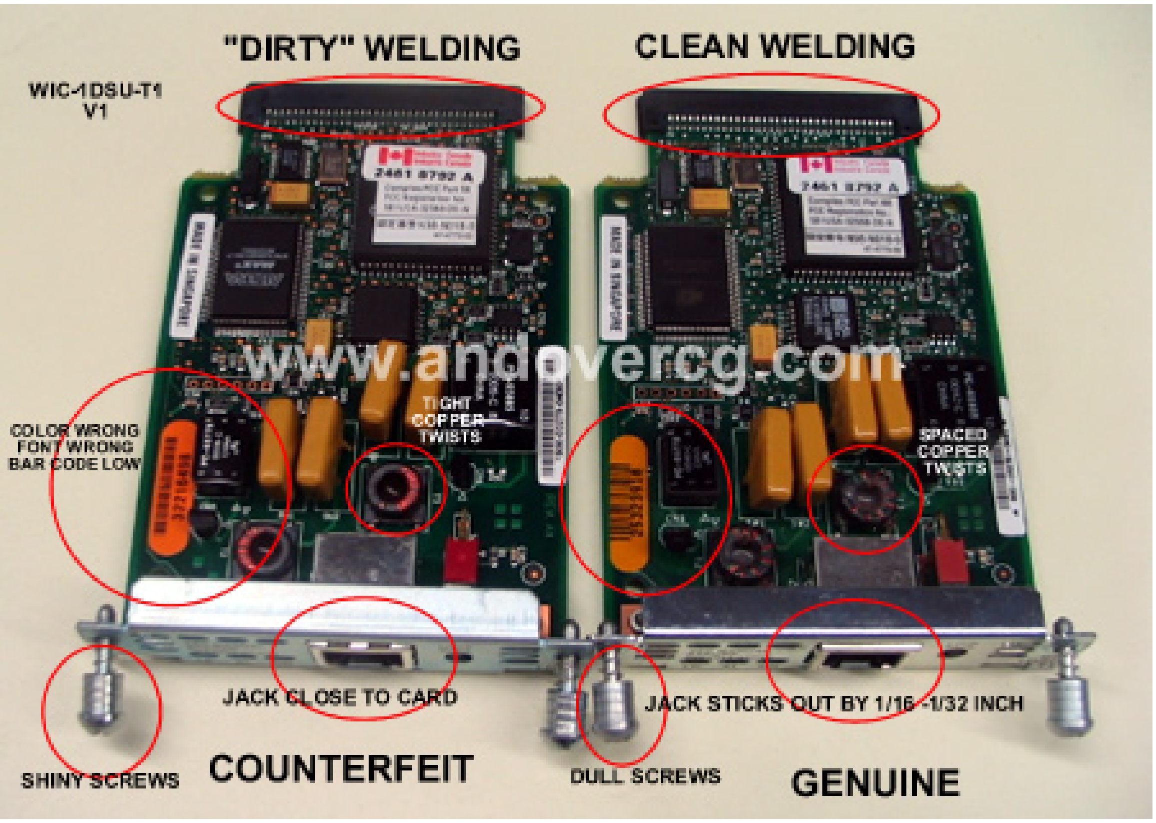


Figure 7. Counterfeit and genuine Cisco card (from: http://www.andovercg.com/services/cisco-counterfeit-wic-1dsu-t1.shtml).

On January 4, 2008, Michael and Robert Edman were charged with trafficking in counterfeit Cisco hardware they had purchased from an individual in China. The counterfeit hardware was then sold through middlemen, and shipped to the United States Marine Corps, Air Force, Federal Aviation Administration, Federal Bureau of Investigation, and several defense contractors, universities and financial institutions

[19]. While this particular incident of counterfeiting has not been shown to be anything other than financially motivated, the implications are clear. With the current global supply chain, it is difficult to discern exactly where components are manufactured and under what conditions. Additionally, intentionally compromised devices, whether for financial gain or for espionage, constitute a threat to national security.

III. JØSANG'S MODEL

In the previous chapter, we provided examples of how vulnerabilities throughout the entire component life cycle were exploited. So, how does one know what components to trust in mission-critical yet unclassified systems? One method is to purchase only accredited components from trusted manufacturers. But, how do we assign trust to these manufacturers? Can opinions on trust be calculated?

In his thesis "Trust and its Ramifications for the DoD Public Key Infrastructure (PKI)," Leonard Gaines analyzed five different trust models with respect to their applicability to modeling the use of PKI within DoD. After reviewing each of the models, he chose to apply Audun Jøsang's model because he felt it was the most comprehensive and had the greatest potential to be practically implemented [20]. We feel that based on Gaines' research, Jøsang's model will provide the best trust model for calculating trust with respect to COTS component manufacturers.

Audun Jøsang presented a model for making trust-based decisions in his paper "Trust-based decision making for electronic transactions" in 1999, in which he focused on using his technique to show how trust in remote agents can be calculated based on trust recommendations from many different sources embedded within public key certificates used in public key cryptography.

In this thesis, we demonstrate the applicability of his method of calculating trust to manufacturers of COTS components. We assume the various government communities will provide trust recommendations expressed mathematically such as in Jøsang's method explained below and then provide an example.

His model can be similarly applied to other areas of the product life cycle, such as the SDLC or applied to software and hardware distribution chain.

A. JØSANG'S MODEL DEFINED

Jøsang expresses "opinions" mathematically as:

$$b + d + u = 1, b, d, u \in [0,1] \tag{1.1}$$

where b, d, and u represent belief, disbelief, and uncertainty, respectively. Uncertainty is used when there is no evidence to support either belief or disbelief. An example demonstrating the uncertainty component can be found in Daniel Ellsberg, "Risk, ambiguity, and the Savage axioms" reproduced below:

> Let us suppose that you confront two urns containing red and black balls, from one of which a ball will be drawn at random. To 'bet on Red' will mean that you choose to draw from Urn I; and that you will receive a prize *a* (say $100) if you draw a red ball and a smaller amount *b* (say $0) if you draw a black. You have the following information: Urn I contains 100 red and black balls, but in ratio entirely unknown to you; there may be from 0 to 100 red balls. In Urn II, you confirm that there are exactly 50 red and 50 black balls.

The probability of drawing a red ball from Urn II is 0.5, since there is an equal number of red balls and black balls in the urn. However, if one was forced to make a bet on the outcome of drawing a red ball from Urn 1, where one does not know the color distribution of the balls, most people will still agree that the probability of drawing a red ball is 0.5, since there are only two different colors in the urn. The value 0.5 is intuitively selected because there are only two possible colors $\theta = \{red, black\}$ and that $|\{red\}| = \frac{1}{2} \mathrm{x} |\theta|$ so the uncertain probability of drawing a red would have been 0.5. If there were five different colors $\theta = \{red, black, blue, yellow, green\}$ then $|\{red\}| = \frac{1}{5} \mathrm{x} |\theta|$ and the uncertain probability of drawing a red would have been 0.2. This concept of calculating the uncertain probability given only the number of states (number of distinct colors in this case) is known as *relative atomicity*, denoted by *a*.

This example illustrates a unique phenomenon where in one case where the distribution is known, and in the other the distribution is unknown, yet they both appear to have the same probability of being selected, 0.5.

In Jøsang's model, $w = (b, d, u, a)$ is an ordered quadruple whose components correspond to belief, disbelief, uncertainty, and relative atomicity, respectively. *w* is defined to be an opinion. An opinion has an ownership which will be designated by a subscript. For example, w_y^A denotes an opinion on proposition *y*, held by agent *A.*

We will use Jøsang quadruples in Chapter IV to manipulate opinions of COTS component manufacturers.

The probability expectation of w, denoted by E(w) is defined by Jøsang to be:

$$E(w) = b + au \tag{1.2}$$

The probability expectation consists of belief, uncertainty, and relative atomicity.

B. SUBJECTIVE LOGIC

Jøsang defines an algebra-based method for manipulating opinions on binary propositions called *subjective logic*. Subjective logic contains the following operators: conjunction, disjunction, negation, recommendation, and consensus. The first three operators are very similar to those of standard Boolean algebra. However, the recommendation and consensus operators are what set *subjective logic* apart from Boolean algebra and standard logic.

1. Conjunction Operator

Given that x and y represent two distinct propositions denoted as $w_x = (b_x, d_x, u_x, a_x)$ and $w_y = (b_y, d_y, u_y, a_y)$ respectively, then the belief that **both** x and y are true is represented by $w_{x \wedge y} = (b_{x \wedge y}, d_{x \wedge y}, u_{x \wedge y}, a_{x \wedge y})$ such that:

1. $b_{x \wedge y} = b_x b_y$
2. $d_{x \wedge y} = d_x + d_y - d_x d_y$
3. $u_{x \wedge y} = b_x u_y + u_x b_y + u_x u_y$
4. $a_{x \wedge y} = \dfrac{b_x u_y a_y + u_x a_x b_y + u_x a_x u_y a_y}{b_x u_y + u_x b_y + u_x u_y}$

and $u_{x\wedge y} \neq 0$. Therefore, $w_{x\wedge y} \equiv w_x \wedge w_y$. Jøsang defined this as a conjunction.

2. Disjunction Operator

Given that x and y represent two distinct propositions denoted as $w_x = (b_x, d_x, u_x, a_x)$ and $w_y = (b_y, d_y, u_y, a_y)$ respectively, then the belief that either x **or** y is true is represented by $w_{x\vee y} = (b_{x\vee y}, d_{x\vee y}, u_{x\vee y}, a_{x\vee y})$ such that:

1. $b_{x\vee y} = b_x + b_y - b_x b_y$

2. $d_{x\vee y} = d_x d_y$

3. $u_{x\vee y} = d_x u_y + u_x d_y + u_x u_y$

4. $a_{x\vee y} = \dfrac{u_x a_x + u_y a_y - b_x u_y a_y - u_x a_x b_y - u_x a_x u_y a_y}{u_x + u_y - b_x u_y - u_x b_y - u_x u_y}$

and $u_{x\vee y} \neq 0$. Therefore, $w_{x\wedge y} \equiv w_x \wedge w_y$. Jøsang defined this as a disjunction.

3. Negation Operator

A negative of an opinion indicates that an opinion is false. This negation is similar to a "NOT" in standard logic. If we let $w_x = (b_x, d_x, u_x, a_x)$ be an opinion about a proposition *x*, then w_x has the following properties:

1. $b_{\neg x} = d_x$

2. $d_{\neg x} = b_x$

3. $u_{\neg x} = u_x$

4. $a_{\neg x} = 1 - a_x$

4. Recommendation Operator

Jøsang also defined a recommendation operator. The recommendation operator allows one to form an opinion about something based on someone else's opinion about it. For example, assume there are two agents, *A* and *B*. *B* has an opinion about a proposition *x*. Jøsang's model allows agent A to form an opinion about proposition x based on his knowledge of agent *B*. Let *A* and *B* be two agents where $w_B^A = (b_B^A, d_B^A, u_B^A, a_B^A)$ represents *A*'s opinion about *B*'s recommendations, and where $w_x^B = (b_x^B, d_x^B, u_x^B, a_x^B)$ is *B*'s opinion about *x* shown as a recommendation to *A*. Let $w_x^{AB} = (b_x^{AB}, d_x^{AB}, u_x^{AB}, a_x^{AB})$, then w_x^{AB} has the following properties:

1. $b_x^{AB} = b_B^A b_x^B$

2. $d_x^{AB} = b_B^A d_x^B$

3. $u_x^{AB} = d_B^A + u_B^A + b_B^A u_x^B$

4. $a_x^{AB} = a_x^B$

w_x^{AB} is called the recommendation between w_B^A and w_x^B, expressing *A*'s opinion about *x* as a result of the recommendation from *B*. Jøsang uses the symbol $\otimes$ to define $w_x^{AB} \equiv w_B^A \otimes w_x^B$.

It can be proved that the recommendation operator is associative but not commutative. This implies that the order in which opinions are combined is significant. The recommendation operator assumes that with a chain including more than one opinion, each opinion is formed independently of other recommendations. This implies that the same entity should not appear more than once in any chain.

We will use Jøsang's recommendation operator in Chapter IV to show how opinions of COTS component manufacturers can help us form opinions on their sub-contractors based on our trust in the component manufacturer.

5. Consensus Operator

Jøsang defines a consensus operator as one that combines opinions on the same proposition in a fair and equal way. For example, suppose two different professors observed the work ethic of a particular student. Each professor might have formed a different opinion about the student dependent on the behavior of the student at that particular time. The Bayesian approach dictates that the consensus operator must then be the opinion that a single professor would have after observing the student during both periods. Jøsang [21] showed the following definition corresponds to this approach and is based on Bayesian calculus.

Let $w_x^A = (b_x^A, d_x^A, u_x^A, a_x^A)$ and $w_x^B = (b_x^B, d_x^B, u_x^B, a_x^B)$ be the opinions of Agents *A* and *B* respectively about the same proposition *x*. $w_x^{A,B} = (b_x^{A,B}, d_x^{A,B}, u_x^{A,B}, a_x^{A,B})$ is the opinion such that:

1. $b_x^{A,B} = (b_x^A u_x^B + b_x^B u_x^A)/k$

2. $d_x^{A,B} = (d_x^A u_x^B + d_x^B u_x^A)/k$

3. $u_x^{A,B} = (u_x^A u_x^B)/k$

4. $a_x^{A,B} = \dfrac{a_x^B u_x^A + a_x^A u_x^B - (a_x^A + a_x^B) u_x^A u_x^B}{u_x^A + u_x^B - 2u_x^A u_x^B}$

Where $k = u_x^A + u_x^B - u_x^A u_x^B$ and where $u_x^A = u_x^B \neq 0$ and $u_x^A = u_x^B \neq 1$. Jøsang calls this the consensus operator between w_x^A and w_x^B and uses the symbol $\oplus$ to represent it and defined it as $w_x^{A,B} \equiv w_x^A \oplus w_x^B$. It indicates an imaginary agent [*A, B*]'s opinion about *x*, as if it represented both.

It can be proved that the consensus operator is both commutative and associative. This implies that the order in which the opinions are combined has no influence on the calculation. As with the recommendation operator, independence of each opinion within the chain is assumed.

We will use Jøsang's consensus operator in Chapter IV to combine multiple independent opinions of COTS component manufacturers into one opinion in an equal and fair manner.

IV. ANALYSIS

We argue Jøsang's model can be used to form an overall opinion about manufacturers of COTS components based upon multiple entities opinions using the subjective logic consensus operator.

Additionally, one could use Jøsang's subject logic recommendation operator to form an opinion on a separate entity's opinion about something or someone. For example, assume there exist two agents *A* and *B* where agent *A* has an opinion about *B*'s trustworthiness and *B* has an opinion on proposition *x*. Jøsang's recommendation model allows agent *A* to form an opinion on proposition *x*. This could be useful for forming an opinion on a subcontractor *x* that works for agent *B*.

In this chapter, we provide examples implementing both Jøsang's consensus and recommendation operators based on opinions provided by trusted third parties.

A. EXAMPLE OF JØSANG'S CONSENSUS OPERATOR

If multiple government agencies formed independent opinions of a fictitious integrated circuit manufacturer, the consensus operator will allow for the formation of one opinion, taking all into account equally and fairly. This should reduce uncertainty.

In this example, we will examine a fictitious integrated circuit manufacture called *Super Good ICs Inc*, using fabricated information provided by three intelligence communities that we will refer to as Intelligence Agencies *A*, *B*, and *C*.

This scenario will operate on three important assumptions. The first assumption is that each of the three intelligence agencies acquired the information that they used to form their opinion independently of the others.

The second assumption is that each of the intelligence agencies has enough knowledge to make an informed opinion about *Super Good ICs Inc.*

The last assumption is that each of the intelligence agencies can be trusted to provide an honest opinion of *Super Good ICs Inc*, e.g., they have not been infiltrated or influenced in some way by another entity.

Recall that the consensus operator in subjective logic (represented by $\oplus$) allows for the combining of multiple opinions about the same proposition into one single opinion taking all into account equally and fairly.

Let $w_{SG}^{A} = (b_{SG}^{A}, d_{SG}^{A}, u_{SG}^{A}, a_{SG}^{A})$ and $w_{SG}^{B} = (b_{SG}^{B}, d_{SG}^{B}, u_{SG}^{B}, a_{SG}^{B})$ be the opinions of agency *A* and *B* respectively about whether *Super Good ICs Inc* is not producing chips that have extra, unauthorized functionality*.* For this example, let $w_{SG}^{A} = (.85,.01,.14,.50)$ and $w_{SG}^{B} = (.80,.05,.15,.50)$ represent belief, disbelief, uncertainty, and atomicity. Atomicity is .5 because there exist only two possibilities, *Super Good ICs* is providing altered chips or it is not.

$w_{SG}^{A} \oplus w_{SG}^{B}$ is calculated using the equations shown Chapter III as follows:

1. $b_{SG}^{A,B} = (b_{SG}^{A} u_{SG}^{B} + b_{SG}^{B} u_{SG}^{A}) / k = (.85*.15 + .80*.14) / (.14 + .15 - .14*.15) = .89$

2. $d_{SG}^{A,B} = (d_{SG}^{A} u_{SG}^{B} + d_{SG}^{B} u_{SG}^{A}) / k = (.01*.15 + .15*.14) / (.14 + .15 - .14*.15) = .08$

3. $u_{SG}^{A,B} = (u_{SG}^{A} u_{SG}^{B}) / k = (.14*.15) / (.14 + .15 - .14*.15) = .08$

4.

$$a_{SG}^{A,B} = \frac{a_{SG}^{B} u_{SG}^{A} + a_{SG}^{A} u_{SG}^{B} - (a_{SG}^{A} + a_{SG}^{B}) u_{SG}^{A} u_{SG}^{B}}{u_{SG}^{A} + u_{SG}^{B} - 2 u_{SG}^{A} u_{SG}^{B}} = \frac{.5*.14 + .5*.15 - (.5 + .5)*.14*.15)}{.14 + .15 - 2*.14*.15} = .5$$

Where $k = u_x^A + u_x^B - u_x^A u_x^B$ and where $u_x^A = u_x^B \neq 0$ and $u_x^A = u_x^B \neq 1$. $w_x^{A,B} = (.89, .08, .08, .5)$ is the consensus of w_x^A and w_x^B is represented by $w_x^A \oplus w_x^B$.

However, if agency *C* has evidence that *Super Good ICs Inc* is using substandard materials likely to cause the chips to fail in an unacceptable period of time and has formed the following opinion $w_{SG}^{C} = (.01, .95, .04, .50)$ then $w_{SG}^{A,B} \oplus w_{SG}^{C}$ becomes:

1.

$b_{SG}^{(A,B),C} = (b_{SG}^{A,B} u_{SG}^{C} + b_{SG}^{C} u_{SG}^{A,B}) / k = (.89*.04 + .01*.08) / (.08 + .04 - .08*.04) = .31$

2.

$d_{SG}^{(A,B),C} = (d_{SG}^{A,B} u_{SG}^{C} + d_{SG}^{C} u_{SG}^{A,B}) / k = (.89*.04 + .04*.08) / (.08 + .04 - .08*.04) = .33$

3. $u_{SG}^{(A,B),C} = (u_{SG}^{A,B} u_{SG}^{C}) / k = (.08*.04) / (.08 + .04 - .08*.04) = .03$

4.

$$a_{SG}^{(A,B),C} = \frac{a_{SG}^{C} u_{SG}^{A,B} + a_{SG}^{A,B} u_{SG}^{C} - (a_{SG}^{A,B} + a_{SG}^{C}) u_{SG}^{A,B} u_{SG}^{C}}{u_{SG}^{A,B} + u_{SG}^{C} - 2 u_{SG}^{A,B} u_{SG}^{C}} = \frac{.5*.08 + .5*.04 - (.5 + .5).08*.04}{.08 + .04 - 2*.08*.04} = .5$$

Where $k = u_x^{A,B} + u_x^C - u_x^{A,B} u_x^C$ and where $u_x^{A,B} = u_x^C \neq 0$ and $u_x^{A,B} = u_x^C \neq 1$. $w_x^{(A,B),C} = (.31, .33, .03, .5)$ is the consensus of $w_x^{A,B}$ and w_x^C is represented by $w_x^{A,B} \oplus w_x^C$.

B. EXAMPLE OF JØSANG'S RECOMMENDATION OPERATOR

The recommendation operator can be used to form an opinion based on someone else's recommendation. For example, if Agency *A* had an opinion on COTS manufacturer *B*, and COTS manufacturer *B* had an opinion on subcontractor *s*, then we can calculate *A*'s opinion on *s* using Jøsang's model.

Let $w_B^A = (b_B^A, d_B^A, u_B^A, a_B^A)$ represents *A*'s opinion about *B*'s recommendations and let $w_s^B = (b_s^B, d_s^B, u_s^B, a_s^B)$ represent *B*'s opinion about *s.*

Given that $w_B^A = (.90, .05, .05, .5)$ and $w_s^B = (.95, .02, .03, .5)$, *A*'s opinion about *s*, represented by $w_s^A = w_B^A \otimes w_s^B$ is calculated as follows:

1. $b_s^{AB} = b_B^A b_s^B = .90 * .95 = .86$

2. $d_s^{AB} = b_B^A d_s^B = .90 * .02 = .02$

3. $u_s^{AB} = d_B^A + u_B^A + b_B^A u_s^B = .05 + .05 + .90 * .03 = .13$

4. $a_s^{AB} = a_s^B = .5$

w_s^A is Agency *A*'s opinion on COTS manufacturer *B*'s subcontractor *s*, formed based on *B*'s opinion and *A*'s "trust" in *B*'s opinion. w_s^A is calculated to be $w_s^A = (.86, .02, .13, .5)$ and is represented by $w_s^A = w_B^A \otimes w_s^B$.

C. USING THE RESULTS OF THE CALCULATED TRUST OPINION

The calculated trust opinion represents the combination of others opinions about certain propositions. How to act on the calculated opinion is subjective. The final decision on how to act on the calculated trust opinion will depend on many factors such as the persons or agencies aversion to risk, the value of the proposition being considered, and the consequences of making a bad decision.

In an automated system, the decision to accept or reject a proposition could be based on pre-defined threshold values established by the organizations policy makers

D. WEAKNESSES WITH JØSANG'S MODEL

Implementing Jøsang's model will require opinions formed using consistent methods able to contend with a variety of situations on a limited number of propositions. Additionally, it will be difficult to verify that the opinions formed by each intelligence agency were formed from independent sources.

1. Forming Good Opinions

Analysts expressing the opinions of their respective organizations (such as from U.S. government agencies, non-government organizations, various private sector companies, and select foreign partners) will be required to form opinions based upon their particular organizations knowledge of an outside organization with respect to a certain proposition. One possible way of assigning belief values is as follows:

- Very strong belief in the proposition (belief value from 1.00 to .90)
- Strong belief in the proposition (belief value from .89 to .70)
- Belief in the proposition (belief value from .69 to .50)

- Dis-belief in the proposition (belief value from .49 to .40)
- Strong dis-belief in the proposition (belief value from .39 to .10)
- Very strong dis-belief in the proposition (belief value from .09 to .0)

V. CROSS DOMAIN INFORMATION SHARING

Chapter IV illustrated a formal model for calculating measures of trust using opinions of parties interested in the COTS components. However, where would one obtain such opinions and how would they be stored? In order to implement Jøsang's trust model to COTS component manufacturers, we would ideally generate opinions about a particular manufacturer from large repositories of information from all available sources, including U.S. government agencies, non-government organizations (NGOs), various private sector companies, and select foreign partners.

This system would need to:

- Share information across many domains spanning organizational boundaries
- Accept input from multiple security levels
- Output information to multiple security levels while protecting the sources and methods used to obtain the information
- Utilize mixed model access control (i.e., the Bell-Padula model on its own is insufficient)
- Enforce domain-specific declassification polices and rules
- Be trustworthy as defined in Chapter I

Currently, there is no formal, efficient, and practical method used to share information spanning multiple domains (e.g., DoD, NGOs, industry, etc) and multiple classifications. Implementation of a cross domain information sharing model will allow cooperation among disparate organizations that might not otherwise know other organizations are working on the same problem.

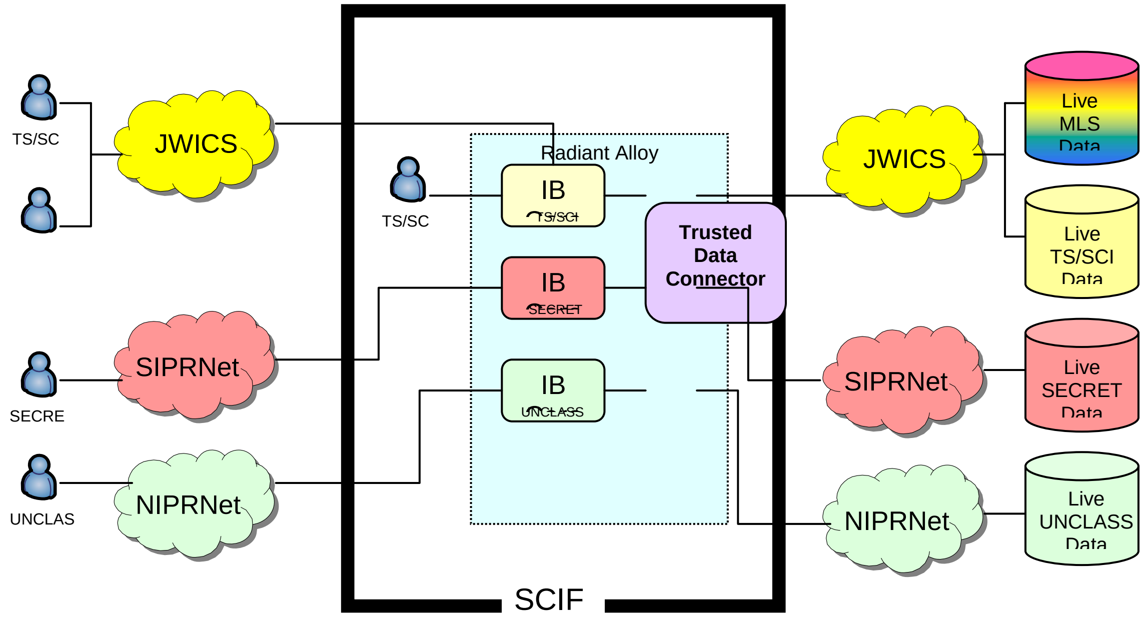


Figure 8. Proposed Radiant Alloy Architecture for High Assurance Systems available from http://www.nps.edu/Research/mdsr/Docs/Vol28Mar08.pdf.

A proposed system that meets some of the requirements is illustrated in Figure 8. Radiant Alloy is expected to provide access using both MLS and Role Based Access Control (RBAC) models. In the proposed model, MLS will be used to divide the various classification domains and RBAC will be used to provide fine grain access control.

The combining of these two methods for access control might result in unexpected weaknesses in the model; Radiant Alloy is already undergoing preliminary C&A. Additionally, convincing the above organizations to share information with organizations that they have not individually vetted might prove difficult.

Radiant Alloy will broker information in an enterprise, multilevel secure (MLS) system using mixed model access control. This system would be tiered in such a way as to prohibit the transfer of classified information to organizations without the required clearances or need to know. Figure 8 illustrates the concept of an Information Broker(IB), an information management controller in the information sharing system which acts as an intermediary between the requestor of the information and the data repositories [22]. The IB should provide the requested data without allowing the user the ability to know or infer the original source of the data (i.e., the "safety property" which guards against information leakage. One method of doing this is to encapsulate the data under the IB's name, maintaining the confidentiality of the requestor and providing repository. The IB requests data through the Trusted Database Connector (TDC) to fulfill user requests. The IB is intended to be a highly reliable component of the information sharing system, able to access various classifications.

For example, if a U.S. intelligence agency has classified information that indicates a particular COTS component manufacturer has added a method of bypassing normal authentication methods to their products, it could indicate so in their opinion in the shared information database. However, if the information was retrieved by an agency without the proper clearance or need to know, the opinion can be "downgraded" by decreasing either the belief or disbelief component of Jøsang's quadruple while increasing the uncertainty component. A similar method is used with respect to the Global Positioning System (GPS) called selective availability (SA). SA introduces intentional, slowly changing random errors of up to a hundred meters in the publicly available navigation signals to prevent mal actors from using GPS based weapons. An encoded GPS signal, the Precise Positioning Service (PPS), which does not contain the SA errors, is primarily used by the DoD [23].

Cover stories can be created to obfuscate the reasoning behind the opinion to un-cleared organizations such as justifying poor opinions based on poor reliability and/or manufacturing defects, poor treatment of factory workers, etc., when in fact it is due to purposeful modifications to components.

VI. CONCLUSIONS AND FUTURE WORK

The reliance on COTS components by DoD may be exploited by well-funded adversaries either by discovering existing un-intentional defects or weaknesses in the components used (since they have access to the same components) or by introducing vulnerabilities at some point during the product lifecycle.

Jøsang's model can be used to calculate trust in COTS manufacturers and their COTS components by providing a systematic and formal way to combine the opinions of multiple entities about the manufacturers and their components. Additionally, it provides the unique ability of calculating the trust in a different entity's opinion based on how much "trust" is placed in the entity submitting the opinion.

To utilize the proposed trust model, DoD will need to implement a cross domain information sharing scheme such as outlined in Chapter V. Populating this system will require various U.S. government agencies, NGOs, various private sector companies, and select foreign partners to submit opinions in a consistent and standard way.

We recommend that DoD only utilize accredited components manufactured by trusted factories tested within the components common and not so common applications in mission-critical and performance-intensive activities.

A. FUTURE WORK

1. Automation

The desired end state of this proposed information sharing and trust model is for the end user to be able to quickly and easily access trust information about component manufacturers suitable for his or her role. This would be most easily accomplished through an automated system.

a. Populating the Model

This proposed method for calculating trust could be more widely applied if the information repository the model uses to calculate opinions is populated by some automated means. Such automation could pull information about individual components from various statistics including: reliability statistics, failure rates, survivability data, expected component life, method of failure (catastrophic vs graceful degradation), etc. An appropriate opinion can then be calculated using the amount of information available as a guide for the belief, disbelief, and uncertainty values.

b. Generating Opinions from the Model

With a populated repository of opinions, the automation of generating the appropriate opinions should be straight forward. However, determining appropriate threshold values on whether to engage in a transaction might prove challenging since different users have different tolerances with respect to risk.

2. Improving upon Jøsang's Model

Is Jøsang's too generalized to be successfully utilized in such a manner? Jøsang's subjective logic trust model allows the calculation of trust with uncertainty and incomplete information.

Jøsang's model could provide too much latitude when forming opinions. He does not provide guidelines on how to formulate opinions or how to assign values to them.

Defining discrete values for opinions will be necessary in order to resolve ambiguity with the implementation of his model. Currently, many Department of the Navy (DoN) projects are tracked using the colors red, yellow, or green to indicate behind schedule, at risk for falling behind schedule, and on/ahead of schedule respectively. It appears obvious that further granularity would be needed to implement this model, but how much more? Are ten distinct divisions (e.g., scale of one to ten) enough to provide the desired precision? Or are more divisions desirable?

3. Implementing the Model

How much would it cost to implement such an information sharing scheme? Who would pay for such a system? Would it be paid for by one organization, or would the costs be shared amongst all of its users?

What policies and/or regulations need to be altered in order to share such information amongst these organizations? Does the proposed Radiant Alloy system meet the requirements set forth in Chapter V? Do any other current or proposed systems meet our system requirements?

Once such a system is established, working groups with representation from all concerned organizations would need to be established to determine how their respective information would be migrated into the new information sharing system.

LIST OF REFERENCES

[1] DoD Acquisition Community Connection, "Secretary of Defense Memorandum: Specifications & Standards - A New Way of Doing Business," https://acc.dau.mil/GetAttachment.aspx?id=32397&pname=file&lang=en-US&aid=6118 (date last accessed September 2008).

[2] R.J. Calantone and A.D. Benedetto, "Performance and Time to Market: Accelerating Cycle Time with Overlapping Stages," *IEEE Transactions on Engineering Management*, vol. 47, p. 232, May 2000.

[3] D. Caffall and B Michael, "Space applications of systems of systems," *Introduction to systems of systems*, pending publication.

[4] Xbox linux project, "Main Page," http://www.xbox-linux.org/wiki/Main_Page (date last accessed September 2008).

[5] Michael Steil, xbox-linux.org "17 Mistakes Microsoft Made in the Xbox Security System," http://www.xbox-linux.org/wiki/ 17_Mistakes_Microsoft_Made_in_the_Xbox_Security_System (date last accessed September 2008).

[6] Committee on National Security Systems, *National Information Assurance Glossary,* CNSS Instruction No. 4009 (revised June 2006). http://www.cnss.gov/Assets/pdf/cnssi_4009.pdf (date last accessed September 2008).

[7] Elias Levy, "Poisoning the Software Supply Chain," *IEEE Security & Privacy*, vol. 1, p. 70, May-June 2003.

[8] Ian Somerville, *Software Engineering*. Boston: Pearson, 2004: pp. 67, 72, 392.

[9] Barry Boehm, "A Spiral Model of Software Development and Enhancement," *ACM SIGSOFT Software Engineering Notes,* 11.4 (1986): pp. 14-24.

[10] John Viega, *Building Secure Software: How to Avoid Security Problems the right way*, Boston: Addison-Wesley 2002: p. 17.

[11] Ian Somerville, *Software Engineering*. Boston: Pearson, 2004: p. 392.

[12] "Moduel 7 Waterfall, Spiral, and 'V' Models," B Michael calls notes for EC4010, Department of Electrical and Computer Science, Naval Postgraduate School, Spring 2008.

[13] IanSomerville, Ian, *Software Engineering*. Boston: Pearson, 2004: p. 72.

[14] "Moduel 7 Waterfall, Spiral, and 'V' Models," B Michael calls notes for EC4010, Department of Electrical and Computer Science, Naval Postgraduate School, Spring 2008.

[15] Sydney Pope, *Trusted Integrated Circuit Strategy,* IEEE Transactions on components and packaging technologies, 31.1 (2008): p. 230.

[16] *Linkages: Manufacturing Trends in Electronic Interconnection Technology,* Nat. Res. Council, 2005, pp. 31-34.

[17] D. Schwaderer, "Innovation Survival - Innovation in Military, Warship Evolution," presented at Secretary of Navy Guest Lecture series, Monterey, CA, Sept 2008. Available from: http://web.mac.com/innovationsurvival/Innovation_Survival/Military.html (date last accessed September 2008).

[18] Redhat Network, "Critical: Openssh Security Update," https://rhn.redhat.com/errata/RHSA-2008-0855.html (date last accessed September 2008).

[19] Federal Bureau of Investigation, "International Initiative Against Traffickers in Counterfeit Network Hardware," http://washingtondc.fbi.gov/dojpressrel/pressrel08/cisco022808.htm (date last accessed September 2008).

[20] Leonard Gaines, "Trust and its Ramifications for the DoD Public Key Infrastructure (PKI)," M.S. thesis, Naval Postgraduate School, Monterey, CA, September 2000.

[21] A. Jøsang, "A metric for trusted systems," *Proceedings of the 21st National Security Conference*, NSA 1998.

[22] Randall Arvay, "Information BrokerArchitectural Pattern to Support Safety of Mixed Model Access control in a Service Oriented Architecture Multilevel Security System," Ph.D. dissertation proposal, Naval Postgraduate School, Monterey, CA, September 2008 p. 2.

[23] United States Coast Guard Navigation Center, "Frequently Asked Questions," http://www.navcen.uscg.gov/faq/gpsfaq.htm#(SA) (date last accessed September 2008).

THIS PAGE INTENTIONALLY LEFT BLANK

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California

3. James B. Michael
   Naval Postgraduate School
   Monterey, California

4. Raymond Buettner
   Naval Postgraduate School
   Monterey, California